05

Viendo el histórico

Transcripción

[00:00] Hola a todos. Bienvenidos de nuevo a este curso de Git. Nos quedamos en el último video en cómo ver el histórico de modificaciones, cada commit que hicimos en nuestro proyecto. Esto lo hacemos solamente ejecutando git log. Este comando muestra muchas cosas.

[00:16] Rápidamente, primero vemos un hash del commit. ¿Qué es un hash? Un hash es una identificación única. Un ID para ese commit. No existen dos commits con el mismo hash. Con esto podemos hacer algunas manipulaciones que vamos a ver más adelante. Segundo, branch. Acá vemos dos palabras.

[00:38] Primero vemos la palabra HEAD. HEAD nos indica dónde estamos parados en este momento en el proyecto. Vemos también, eso de aquí, el HEAD no es tan importante ahora, en este momento. Master es la branch o la rama. Esta parte sí es importante pero por ahora no lo vamos a ver. Lo vamos a ver también más adelante en otro video.

[01:07] Luego tenemos el autor, que acá aparece mi nombre y mi email. Luego tenemos la fecha en que fue hecho ese commit y luego el mensaje que hemos colocado. Lo mismo con ese segundo commit. Ahora, ¿cómo sabe Git cuál es mi nombre y mi email? Bueno, anteriormente yo les había hablado sobre el git config y ahora lo vamos a ver un poco más en detalle.

[01:34] ¿Cómo hacer para configurar mi email y mi nombre? Lo que tenemos que hacer es colocar git config y acá existen dos parámetros para colocar. Puede ser --local, que esto significa que las configuraciones que vamos a crea

van a ser solo para este repositorio en este proyecto y existe la configuración --global, que esto es para todos los proyectos que vamos a hacer de ahora en adelante en nuestra computadora.

[02:01] Entonces en este caso voy a usar --local y vamos a colocar user.name para definir cuál es nuestro nombre. Luego de user.name hacemos espacio, colocamos comillas y colocamos nuestro nombre. Voy a cambiarlo acá a Bruno Fernández. Cierro comillas, doy enter y ahora si yo ejecuto git config, espacio, user.name, doy enter, me va a decir mi nombre.

[02:33] ¿Qué otra cosa puedo configurar? Puedo poner mi email. Vamos a usar la flechita de nuestro teclado, la flechita para arriba, vamos a hacer una vez para ir a este commit, dos veces para ir al anterior y así sucesivamente. En este caso, vamos a borrar todo esto y vamos a cambiar por user.email para definir ese email que aparece acá. Entonces vamos a colocar user.email, espacio, abrimos comillas y colocamos nuestro email.

[03:04] Yo ya lo tengo colocado pero ustedes colocan ahí su email normalmente. Y si quieren ver si su email está bien puesto, colocan user.email, enter y ahí aparece su email. Perfecto, ahora, volviendo para git log quizás se pregunten, ¿existe otra forma o formato para ver ese git log? Sí, existen infinidad de formas de ver esas informaciones. Una muy común es visualizar todos los commits donde cada uno de ellos ocupen solo una línea.

[03:37] Voy a hacer un clear aquí, para limpiar la pantalla para que no tenga tanta información, damos enter y ahí vamos a utilizar el comando git, espacio, lógica, espacio, --oneline. Una línea en inglés. Con esto vemos que primero al principio tenemos un resumen del hash, los primeros caracteres del hash, tenemos en cuál branch fue ejecutado y tenemos acá la información que nosotros colocamos en el mensaje, en el commit.

[04:12] ¿Pero si en vez de ver menos, quiero ver más? ¿Qué tal si quiero ver las modificaciones de los archivos? Lo podemos hacer usando git log, espacio, -p.

Enter y acá vemos, este fue nuestro primer commit, nos dice todas las informaciones que ya habíamos visto anteriormente, pero además de eso nos hace lo que se llama un div que nos va diciendo cuáles fueron las modificaciones que fueron realizadas en ese commit.

[04:37] En este caso, el último commit lo que nos dice fue que se quitó esa línea y se agregó esa línea con acento. Si damos acá enter para continuar hacia abajo vemos que el commit anterior a ese último es este que vemos que esta todo en verde. Todo en verde significa que se agregaron todas esas líneas. ¿Y por qué?

[05:00] Porque simplemente ese archivo fue el primer archivo que colocamos dentro de nuestro Git, entonces todas las líneas son nuevas para Git. Para salir de esa parte del git log -p pueden hacer :q, así, voy a mostrar acá de nuevo: git log, espacio -p, enter, damos enter, enter, enter, enter y acá podemos hacer dos puntos y la letra Q y salimos de ese editor.

[05:29] ¿Puedo hacer más cosas? Existe una infinidad de formas de mostrar nuestro histórico. Voy a dejarles un link de algunas opciones que podemos usar. En este caso estoy usando Devhints.io/git-log. Acá nos muestra muchas cosas que podemos agregar a nuestro git log, por ejemplo podemos colocar el número máximo de commits que queremos ver, si queremos buscar por un mensaje en particular dentro de nuestros commits.

[05:58] O por ejemplo podemos colocar un pretty, que es una forma más customizada, más de acuerdo a lo que nosotros queremos mostrar. En este caso vamos a utilizar este pretty para dar un ejemplo: clic derecho + copiar, volvemos a Git Bash, entonces colocamos git, espacio, log, espacio, y ahora clic derecho + pegar. Podemos utilizar también el comando "Shift + Insert" para colocar lo que hemos copiado acá. Entonces en este caso coloco ahí, damos un Enter.

[06:34] Y vean que en este caso, solamente estoy viendo los hash de cada uno de los commits, nada más. Vamos a customizar o a mejorar esto. Volvemos a

nuestra página, acá tenemos un link para las cosas que podemos colocar dentro del pretty. Y acá vemos con el %h vemos un commit abreviado, para que no ocupe tanto espacio en la pantalla.

[06:59] Un %s nos pone por ejemplo cuál fue el mensaje. Entonces vamos a hacer eso. Vamos a colocar ese H y ese S. Vamos a dar un clear para limpiar un poco la pantalla, usamos la flechita de arriba del teclado para ir a los comandos anteriores que hemos colocado. Voy a colocar H, espacio.

[07:19] Voy a colocar ahora %s, doy enter y acá vemos el hash abreviado y el mensaje. Ahora, ¿qué tal si quiero agregar el email? El email de la persona por ejemplo. Bueno, %+A+E. Vamos a usar la flechita para arriba y acá agregamos %ae. Damos enter y acá vemos que luego del mensaje aparece el email.

[07:54] Yo les voy a dejar el link de esta página en los comentarios para que puedan jugar y divertirse ahí un poco más con Git Log y Git Log Pretty. Ya mostramos de forma linda el log o histórico de nuestro repositorio pero ahora quiero hablar sobre otra cosa. Solo para dar un ejemplo voy a mostrarles que acá estoy usando Visual Studio Code.

[08:18] Es un editor bien simple de código y ustedes pueden utilizar el editor que quieran o un IDE por ejemplo. Ahora, ¿qué ocurre si ese IDE nos crea algunas configuraciones que no son realmente necesarias para el proyecto sino que son simplemente del IDE?

[08:36] Bueno, vamos a simular eso. Acá yo voy a crear una nueva carpeta, P la voy a llamar, enter, y acá adentro voy a crear un nuevo archivo, por ejemplo un archivo que es de configuración. ¿Qué ocurre si yo estos archivos no los quiero realmente en mi proyecto? Entonces, ¿cómo hago para que Git ignore esos archivos? Eso lo vamos a hacer con Git ignore, y lo vamos a ver en el próximo video.